Label-Embedding for Attribute-Based Classification

Zeynep Akata[a,b], Florent Perronnin[a], Zaid Harchaoui[b] and Cordelia Schmid[b]
[a] Computer Vision Group*, XRCE, France [b] LEAR†, INRIA, France

Abstract

*Attributes are an intermediate representation, which enables parameter sharing between classes, a must when training data is scarce. We propose to view attribute-based image classification as a label-embedding problem: each class is embedded in the space of attribute vectors. We introduce a function which measures the compatibility between an image and a label embedding. The parameters of this function are learned on a training set of labeled samples to ensure that, given an image, the correct classes rank higher than the incorrect ones. Results on the Animals With Attributes and Caltech-UCSD-Birds datasets show that the proposed framework outperforms the standard Direct Attribute Prediction baseline in a zero-shot learning scenario. The label embedding framework offers other advantages such as the ability to leverage alternative sources of information in addition to attributes (*e.g. *class hierarchies) or to transition smoothly from zero-shot learning to learning with large quantities of data.*

1. Introduction

We consider the image classification problem: given an image, we wish to annotate it with one (or multiple) class label(s) describing its visual content. Image classification is a prediction task where the goal is to learn from labeled data a function $f : \mathcal{X} \rightarrow \mathcal{Y}$ which maps an input x in the space of images $\mathcal{X}$ to an output y in the space of class labels $\mathcal{Y}$. In this work, we are especially interested in the case where we have *no (positive) labeled samples* for some of the classes and still wish to make a prediction. This problem is generally referred to as zero-shot learning [17, 25, 16, 8].

A solution to zero-shot learning which has recently gained in popularity in the computer vision community consists in introducing an intermediate space $\mathcal{A}$ referred to as *attribute* layer [16, 8]. Attributes correspond to high-level properties of the objects which are *shared* across multiple classes, which can be detected by machines and which can be understood by humans. As an example, if the classes correspond to animals, possible attributes include "has paws", "has stripes" or "is black". The traditional attribute-based prediction algorithm requires learning one classifier per attribute. To classify a new image, its attributes are predicted using the learned classifiers and the attribute scores are combined into class-level scores. This two-step strategy is referred to as Direct Attribute Prediction (DAP) in [16].

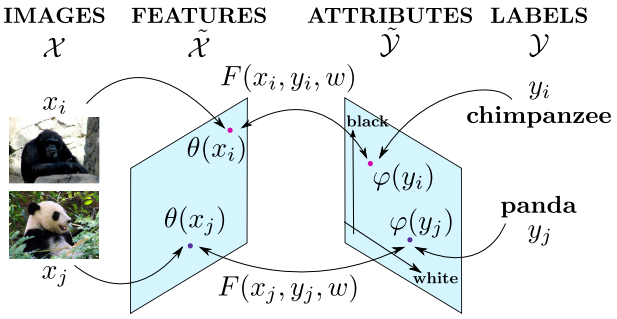


Figure 1. Much work in computer vision has been devoted to image embedding (left): how to extract suitable features from an image? We focus on *label embedding* (right): how to embed class labels in a Euclidean space? We use attributes as side information for the label embedding and measure the "compatibility"' between the embedded inputs and outputs with a function F.

We note that DAP suffers from several shortcomings. First, a two-step prediction process goes against the philosophy which advocates solving a problem directly rather than indirectly through intermediate problems. In other words, since attribute classifiers are learned independently of the end-task they might be optimal at predicting attributes but not necessarily at predicting classes. Second, we would like an approach which can improve incrementally as new training samples are provided, *i.e.* which can perform zero-shot prediction if no labeled samples are available for some classes, but which can also leverage new labeled samples for these classes as they become available. While DAP makes sense for zero-shot learning, it is not straightforward to extend it to such an incremental learning scenario. Third, while attributes can be a useful source of prior information, other sources of information could be leveraged for zero-shot learning. For instance, semantic hierarchies such as

*The Computer Vision Group at XRCE is partially funded by the ANR project FIRE-ID.
†The LEAR team at INRIA is partially funded by the European integrated project AXES.

Wordnet[1] can bring useful information. Indeed, images of classes which are close in a semantic hierarchy are usually more similar than images of classes which are far [6]. It is not straightforward to design an efficient way to incorporate these additional sources of information into DAP.

Various solutions have been proposed to address each of these problems separately (see section 2). However, we do not know of any existing solution which addresses all of them in a principled manner. This paper proposes such a solution by making use of the *label embedding* framework. We underline that, while there is an abundant literature in the computer vision community on image embedding (how to describe an image?) much less work has been devoted in comparison to label embedding in the $\mathcal{Y}$ space (how to describe a class?). We embed each class $y \in \mathcal{Y}$ in the space of attribute vectors and thus refer to our approach as *Attribute Label Embedding* (ALE). We introduce a function which measures the "compatibility" between an image x and a label y (see Figure 1). The parameters of this function are learned on a training set of labeled samples to ensure that, given an image, the correct class(es) rank higher than the incorrect ones. Given a test image, recognition consists in searching for the class with the highest compatibility.

ALE addresses in a principled fashion all three problems mentioned previously. First, we do not solve any intermediate problem and learn the model parameters to optimize directly the class ranking. We show experimentally that ALE outperforms DAP in the zero-shot setting. Second, if available, labeled samples can be added incrementally to update the embedding. Third, the label embedding framework is generic and not restricted to attributes. Other sources of prior information can be combined with attributes.

The paper is organized as follows. In the next section, we review related work. In section 3, we introduce ALE. In section 4 we present experimental results on two public datasets: Animals with Attributes (AWA) [16] and Caltech-UCSD-Birds (CUB) [37]. Finally, we draw conclusions.

2. Related Work

We now review related work on attributes, zero-shot learning and label embedding (three research areas which strongly overlap) with an emphasis on the latter.

2.1. Attributes

Attributes have been used to describe images [9, 8], to generate captions [14, 23], for retrieval [15, 33, 7] and classification [16, 8, 38, 39, 19, 31, 21]. It has been proposed to improve the standard DAP model to take into account the correlation between attributes or between attributes and classes [38, 39, 43, 19]. However, these models have limitations. Wang and Forsyth [38] assume that *images* are labeled with both classes and attributes. In our work we only assume that *classes* are labeled with attributes, which requires significantly less hand-labeling of the data. Mahajan *et al*. [19] use transductive learning and, therefore, assume that the test data is available as a batch, a strong assumption we do not make. Yu and Aloimonos's topic model [43] is only applicable to bag-of-visual-word image representations and, therefore, cannot leverage recent state-of-the-art image features such as the Fisher Vector [26, 4]. Finally the latent SVM framework of Wang and Mori [39] is not applicable to zero-shot learning.

[1] http://wordnet.princeton.edu/

2.2. Zero-shot learning

Zero-shot learning requires the ability to transfer knowledge from classes for which we have training data to classes for which we do not. Possible sources of prior information include attributes [16, 8, 25, 28, 27], semantic class taxonomies [27, 22] or text features [25, 28, 27]. Other sources of prior information can be used for special purpose problems. For instance, Larochelle *et al*. [17] encode characters with 7×5 pixel representations. It is unclear, however, how such an embedding could be extrapolated to the case of generic visual categories. Finally, few works have considered the problem of transitioning from zero-shot to "few-shots" learning [43, 31].

2.3. Label embedding

In computer vision, a vast amount of work has been devoted to input embedding, *i.e*. how to represent an image? This includes works on patch encoding (see [4] for a recent comparison), on kernel-based methods [32] with a recent focus on explicit embeddings [20, 35], on dimensionality reduction [32] and on compression [13, 30, 36]. Comparatively, much less work has been devoted to label embedding.

Provided that the embedding function φ is chosen correctly, label embedding can be an effective way to share parameters between classes. Consequently, the main applications have been multiclass classification with many classes [1, 40, 41, 2] and zero-shot learning [17, 25]. We now provide a taxonomy of embeddings. While this taxonomy is valid fot both input θ and output embeddings φ, we focus here on output embeddings. They can be (i) fixed and data-independent, (ii) learned from data, or (iii) computed from side information.

Data-independent embeddings. Kernel dependency estimation [42] is an example of a strategy where φ is data-independent and defined implicitly through a kernel in the $\mathcal{Y}$ space. Another example is the compressed sensing approach of Hsu *et al*. [12], where φ corresponds to random projections.

Learned embeddings. A possible strategy consists in learning directly an embedding from the input to the output (or from the output to the input) as is the case of re-

gression [25]. Another strategy consists in learning jointly θ and φ to embed the inputs and outputs in a common intermediate space $\mathcal{Z}$. The most popular example is Canonical Correlation Analysis (CCA) [11], which maximizes the correlation between inputs and outputs. Other strategies have been investigated which maximize directly classification accuracy, including the nuclear norm regularized learning of Amit *et al.* [1] or the WSABIE algorithm of Weston *et al.* [41].

Embeddings derived from side information. There are situations where side information is available. This setting is particularly relevant when little training data is available, as side information and the derived embeddings can compensate for the lack of data. Side information can be obtained at an image level [8] or at a class level [16]. We focus on the latter setting which is more practical as collecting side information at an image level is more costly. Side information may include "hand-drawn" descriptions [17], text descriptions [8, 16, 25] or class taxonomies [40, 2].

In our work, we focus on embeddings derived from side information but we also consider the case where they are learned from labeled data, using side information as a prior.

3. Learning with attributes as label embedding

Given a training set $\mathcal{S} = \{(x_n, y_n), n = 1 \dots N\}$ of input/output pairs with $x_n \in \mathcal{X}$ and $y_n \in \mathcal{Y}$ the goal of prediction is to learn a function $f : \mathcal{X} \to \mathcal{Y}$ by minimizing an empirical risk of the form $\frac{1}{N}\sum_{n=1}^{N} \Delta(y_n, f(x_n))$ where $\Delta : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$ measures the loss incurred from predicting $f(x)$ when the true label is y. In what follows, we focus on the 0/1 loss: $\Delta(y, z) = 0$ if $y = z$, 1 otherwise. In machine learning, a common strategy is to use *embedding functions* $\theta : \mathcal{X} \to \tilde{\mathcal{X}}$ and $\varphi : \mathcal{Y} \to \tilde{\mathcal{Y}}$ for the inputs and outputs and then to learn on the transformed input/output pairs.

In this section, we first describe our model, *i.e.* our choice of f. We then explain how to leverage attributes to compute label embeddings. We also discuss how to learn the model parameters. Finally, we show that the label embedding framework is generic enough to accommodate for other sources of side information.

3.1. Model

Figure 1 illustrates our model. As is common in structured prediction [34], we introduce a compatibility function $F : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$ and define the prediction function f as follows:

$$f(x; w) = \arg\max_{y \in \mathcal{Y}} F(x, y; w) \tag{1}$$

where w denotes the model parameter vector of F and $F(x, y; w)$ measures how compatible is the pair (x, y) given w. It is generally assumed that F is linear in some combined feature embedding of inputs/outputs $\psi(x, y)$:

$$F(x, y; w) = w' \psi(x, y) \tag{2}$$

and that the joint embedding ψ can be written as the tensor product between the image embedding $\theta : \mathcal{X} \to \tilde{\mathcal{X}} = \mathbb{R}^D$ and the label embedding $\varphi : \mathcal{Y} \to \tilde{\mathcal{Y}} = \mathbb{R}^E$:

$$\psi(x, y) = \theta(x) \otimes \varphi(y) \tag{3}$$

and $\psi(x, y) : \mathbb{R}^D \times \mathbb{R}^E \to \mathbb{R}^{DE}$. In this case w is a DE-dimensional vector which can be reshaped into a $D \times E$ matrix W. Consequently, we can rewrite $F(x, y; w)$ as a bilinear form:

$$F(x, y; W) = \theta(x)' W \varphi(y). \tag{4}$$

Other compatibility functions could have been considered. For example, the function:

$$F(x, y; W) = -||\theta(x)' W - \varphi(y)||^2 \tag{5}$$

is typically used in regression problems. If D and E are large, it might be advantageous to consider a low-rank decomposition $W = U'V$ to reduce the number of parameters. In such a case, we have:

$$F(x, y; U, V) = (U\theta(x))' \, (V\varphi(y)) \, . \tag{6}$$

CCA [11] or WSABIE [41] rely, for example, on such a decomposition.

3.2. Attribute label embedding

We now consider the problem of computing label embeddings $\varphi^{\mathcal{A}}$ from attributes which we refer to as Attribute Label Embedding (ALE). We assume that we have C classes, *i.e.* $\mathcal{Y} = \{1, \dots, C\}$ and that we have a set of E attributes $\mathcal{A} = \{a_i, i = 1 \dots E\}$ to describe the classes. We also assume that we are provided with an association measure $\rho_{y,i}$ between each attribute a_i and each class y. These associations may be binary or real-valued if we have information about the association strength. In this work, we focus on binary relevance although one advantage of the label embedding framework is that it can easily accommodate real-valued relevances. We embed class y in the E-dim attribute space as follows:

$$\varphi^{\mathcal{A}}(y) = [\rho_{y,1}, \dots, \rho_{y,E}] \tag{7}$$

and denote $\Phi^{\mathcal{A}}$ the $E \times C$ matrix of attribute embeddings which stacks the individual $\varphi^{\mathcal{A}}(y)$'s. We note that in equation (4) the image and label embeddings play symmetric roles. It can make sense to normalize the output vectors $\varphi^{\mathcal{A}}(y)$. In the experiments, we consider among others mean-centering and ℓ_2-normalization.

Also, in the case where attributes are redundant, it might be advantageous to decorrelate them. In such a case, we make use of the compatibility function (6). The matrix V may be learned from labeled data jointly with U. As a simpler alternative, it is possible to first learn the decorrelation, *e.g.* by performing a Singular Value Decomposition (SVD) on the $\Phi^{\mathcal{A}}$ matrix, and then to learn U. We will study the effect of attribute decorrelation in our experiments.

3.3. Parameter learning

We now turn to the estimation of the model parameters w from a labeled training set $\mathcal{S}$. The simplest learning strategy is to maximize directly the compatibility between the input and output embeddings $\frac{1}{N}\sum_{n=1}^{N} F(x_n, y_n; W)$, with potentially some constraints and regularizations on W. This is exactly the strategy adopted in regression or CCA. However, such an objective function does not optimize directly our end-goal which is image classification. Therefore, we draw inspiration from the WSABIE algorithm [41] which learns jointly image and label embeddings from data to optimize classification accuracy. The *crucial difference between WSABIE and ALE is the fact that the latter uses attributes as side information*.

We first review briefly the WSABIE objective function [41] and then explain how we adapt it to (i) zero-shot learning with side information and (ii) learning with few (or more) examples with side information. We then mention the optimization of our objective functions. In what follows, Φ is the matrix which stacks the embeddings $\varphi(y)$.

WSABIE [41]. Let $\mathbb{1}(u) = 1$ if u is true and 0 otherwise. Let $\ell(x_n, y_n, y) = \Delta(y_n, y) + F(x_n, y; W) - F(x_n, y_n; W)$ and let $r_\Delta(x_n, y_n) = \sum_{y \in \mathcal{Y}} \mathbb{1}(\ell(x_n, y_n, y) > 0)$ be an upper-bound on the rank of label y_n for image x_n. WSABIE considers the following ranking objective:

$$R(\mathcal{S}; W, \Phi) = \frac{1}{N}\sum_{n=1}^{N} \gamma_{r_\Delta(x_n, y_n)} \sum_{y \in \mathcal{Y}} \max\{0, \ell(x_n, y_n, y)\} \tag{8}$$

where γ_k is a decreasing function of k. Maximizing (8) enforces correct labels to rank higher than incorrect ones. A decreasing γ_k ensures that more importance is given to the top of the ranking list, a desirable property. Weston *et al*. optimize objective (8) with respect to W and Φ with constraints on the norms of W and Φ. In WSABIE, the label embedding space dimensionality is a parameter to tune.

Zero-shot learning. We adapt the WSABIE objective to zero-shot learning. In such a case, we cannot learn Φ from labeled data (contrary to WSABIE) but rely on side information. Therefore, the matrix Φ is fixed and set to $\Phi^{\mathcal{A}}$. We only optimize the objective (8) with respect to W. We note that, when Φ is fixed and only W is learned, the objective (8) is closely related to the (unregularized) structured SVM (SSVM) objective [34]:

$$\frac{1}{N}\sum_{n=1}^{N} \max_{y \in \mathcal{Y}} \ell(x_n, y_n, y) \tag{9}$$

The main difference is the loss function. SSVM uses a multiclass objective function, and it only considers rank 1 while WSABIE considers all ranks in a weighted fashion.

Few-shots learning. We now adapt the WSABIE objective to the case where we have labeled data and side information. In such a case, we want to learn the class embeddings using as prior information $\Phi^{\mathcal{A}}$. We therefore add to the objective (8) a regularizer:

$$R(\mathcal{S}; W, \Phi) + \frac{\mu}{2}||\Phi - \Phi^{\mathcal{A}}||^2 \tag{10}$$

and optimize jointly with respect to W and Φ. Note that the previous equation is somewhat reminiscent of the ranking model adaptation of [10].

Optimization. As for the optimization, both in the zero-shot and few-shots learning, we follow [41] and use Stochastic Gradient Descent (SGD). This is a fast procedure which samples both training samples and classes.

3.4. Beyond attributes

While attributes make sense in the label embedding framework, we note that label embedding is more general and can accommodate for other sources of side information. The canonical example is that of structured learning with a taxonomy of classes [34]. Assuming that classes are organized in a tree structure, meaning that we have an ordering operation $\prec$ in $\mathcal{Y}$, we can define $\xi_{y,z} = 1$ if $z \prec y$ or $z = y$. The hierarchy embedding $\varphi^{\mathcal{H}}(y)$ can be defined as the C dimensional vector:

$$\varphi^{\mathcal{H}}(y) = [\xi_{y,1}, \ldots, \xi_{y,C}]. \tag{11}$$

We later refer to this embedding as Hierarchy Label Embedding (HLE) and we compare $\varphi^{\mathcal{A}}$ and $\varphi^{\mathcal{H}}$ as sources of prior information in our experiments. In the case where classes are not organized in a tree structure but form a graph, then other types of embeddings could be used, for instance by performing a kernel PCA on the commute time kernel [29].

Different embeddings can be easily combined in the label embedding framework, *e.g*. through simple concatenation of the different embeddings or through more complex operations such as a CCA of the embeddings. This is to be contrasted with DAP which cannot accommodate so easily other sources of prior information.

4. Experiments

The experimental setup is described in section 4.1. In section 4.2, we present zero-shot learning experiments. In section 4.3, we go beyond zero-shot learning and consider the case where we have labeled training data for all classes.

4.1. Experimental setup

Datasets. We report results on two public datasets. Animal With Attributes (AWA) [16] contains roughly 30,000 images of 50 animal classes. Each class was annotated with 85 attributes by 10 students [24] and the result was binarized. CUB-200-2011 [37] contains roughly 11,800 images of 200 bird classes. Each class is annotated with 312 binary

	RR	Multi	Rank
AWA	30.7	**37.7**	37.4
CUB	13.3	16.4	**18.0**

Table 1. Comparison of different objective functions: ridge regression (RR), the standard SSVM based on multi-class classification (Multi) and the ranking objective of section 3.3 (Rank).

attributes derived from a bird field guide website. Hence, there is a significant difference in the number and quality of attributes between the two datasets. On both datasets, to be consistent throughout our zero-shot and few-shots experiments, we use for each class half of the data for training and the other half for testing. We report results in terms of top-1 accuracy (in %) averaged over the classes.

Features. We extract 128-dim SIFT descriptors [18] and 96-dim color descriptors [5] from regular grids at multiple scales. Both of them are reduced to 64-dim using PCA. These descriptors are then aggregated into an image-level representation using the Fisher Vector (FV) [26] which was shown to be a state-of-the-art patch encoding technique in [4]. Using Gaussian Mixture Models with 256 Gaussians, we compute one SIFT FV and one color FV per image and concatenate them into a 65,536-dim FV which we compress with PQ [13]. These FVs are our image embeddings $\theta(x)$[2].

4.2. Zero-shot learning

We now evaluate the proposed ALE in the zero-shot setting. For AWA, we use the standard zero-shot setup which consists in learning parameters on 40 classes and evaluating accuracy on 10 classes. In these experiments, we use the "train" part of the 40 learning classes to learn and cross-validate the model parameters. We use the "test" part of the 10 evaluation classes to measure accuracy. For CUB, we use 150 classes for learning (using again the "train" part for training and cross-validation) and 50 for evaluation (using only their "test" part).

We answer the following questions. What is the best way to learn the parameters of our model? What is the best way to encode/normalize the attribute embeddings? How does ALE compare to DAP? Do we still learn human-interpretable attribute classifiers? How do attributes compare to a class hierarchy as prior information?

Comparison of learning frameworks. We first compare the learning framework introduced in section 3.3 with two baselines. The first baseline is Ridge Regression (RR) which was used in [25] to map input features to output attribute labels. In a nutshell, RR consists in optimizing a regularized quadratic loss for which there exists a closed form formula. The second baseline is the standard SSVM multiclass objective function (see section 3.3). For these experiments, the attribute vectors are encoded in a binary fashion (using $\{0, 1\}$) and ℓ_2-normalized.

[2] On AWA we also ran experiments with the features provided with the dataset. We obtained similar results to those of FVs. The advantage of the FVs is that they work well with linear classifiers. Therefore, we only report results with FVs.

	$\{0, 1\}$	$\{-1, +1\}$	mean-centered
AWA			
no ℓ_2-norm	37.3	37.3	36.9
with ℓ_2-norm	**37.4**	37.3	36.6
CUB			
no ℓ_2-norm	16.3	17.9	16.5
with ℓ_2-norm	**18.0**	17.9	17.2

Table 2. Comparison of different attribute embeddings: $\{0, 1\}$ embedding, $\{-1, +1\}$ embedding and mean-centered embedding, with and without ℓ_2-norm.

Table 1 shows that the multiclass and ranking objectives perform on par. They outperform significantly ridge regression. This is not surprising, since the two former objective functions are more closely related to our end goal which is classification. In what follows, we always use the ranking framework to learn the parameters of our model.

Comparison of attribute embeddings. We compare different approaches to embed attributes. We experiment with a $\{0, 1\}$ embedding, a $\{-1, +1\}$ embedding and a mean-centered embedding (*i.e.* starting from the $\{0, 1\}$ embedding, we compute the mean over all learning classes and subtract it). Underlying the $\{0, 1\}$ embedding is the assumption that the presence of the same attribute in two classes should contribute to their similarity, but not its absence[3]. Underlying the $\{-1, 1\}$ embedding is the assumption that the presence or the absence of the same attribute in two classes should contribute equally to their similarity. As for mean-centered attributes, they take into account the fact that some attributes are more frequent than others. For instance, if an attribute appears in almost all classes, then in the mean-centered embedding, its absence will contribute more to the similarity than its presence[4]. We also experimented with ℓ_2-normalization of the embedded attribute vectors. The ℓ_2-normalization enforces that each class is closest to itself according to the dot-product similarity. From the results in Table 2, we conclude that all embeddings perform similarly, especially after ℓ_2-norm. In what follows, we make use of the simple $\{0, 1\}$ embedding with ℓ_2-norm.

Comparison of ALE and DAP. We now compare the proposed framework to the DAP baseline of Lampert *et al.* [16]. In DAP, given a new image x, we assign it to the class y with

[3] Here we assume a dot-product similarity between attribute embeddings which is consistent with our linear compatibility function (4).

[4] This is similar to an IDF effect in TF-IDF encoding.

	Obj. pred.		Att. pred.	
	DAP	ALE	DAP	ALE
AWA	36.1	**37.4**	**71.9**	65.7
CUB	10.5	**18.0**	**61.8**	60.3

Table 3. Comparison of the DAP baseline [16] with the proposed Attribute Label Embedding (ALE) approach. Left 2 columns: object classification accuracy (top-1 in %) on the 10 AWA and 50 CUB evaluation classes. Our DAP results on AWA are lower than those reported in [16] because we use only half of the data to train the attribute classifiers. Right 2 columns: attribute prediction accuracy (AUC in %) on the 85 AWA and 312 CUB attributes.

the highest posterior probability:

$$p(y|x) \propto \prod_{e=1}^{E} p(a_e = \rho_{y,e}|x) \tag{12}$$

where $\rho_{y,e}$ is the association measure between attribute a_e and class y, and $p(a_e = 1|x)$ is the probability that image x contains attribute e. We train for each attribute one linear classifier on the FVs. We use a (regularized) logistic loss which provides an attribute classification accuracy similar to the SVM but with the added benefit that its output is already a probability.

From the results in Table 3 (left columns), we can see that the proposed framework performs slightly better than DAP on AWA and significantly better on CUB. Hence, our approach seems to be more beneficial when the attribute quality is higher. The benefits of our approach with respect to DAP are the fact that our objective function optimizes a ranking objective which is closely related to the classification end-goal and the fact that we take into account implicitly the correlation between classes.

Attribute interpretability. In ALE, each column of W can be interpreted as an attribute classifier and $\theta(x)'W$ as a vector of attribute scores of x. However, one major difference with DAP is that we do not optimize for attribute classification accuracy. This might be viewed as a disadvantage of our approach as we might loose interpretability, an important property of attribute-based systems when, for instance, one wants to include a human in the loop [3, 37]. We therefore measured the attribute prediction accuracy of DAP and ALE. For each attribute, following [16], we measure the AUC on the "test" set of the evaluation classes and report the mean.

Results are shown in Table 3 (right columns). As expected, the attribute prediction accuracy of DAP is higher than that of our approach. Indeed, DAP optimizes directly attribute-classification accuracy. However, the AUC for the proposed approach is still reasonable, especially on CUB (only 1.5% drop). Thus, our learned attribute classifiers should still be interpretable. We show qualitative results on AWA in Figure 2.

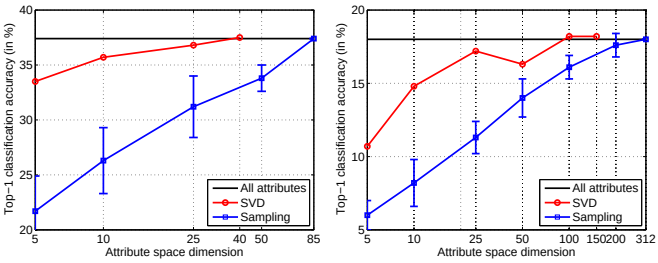


Figure 3. Classification accuracy on AWA (left) and CUB (right) as a function of the label embedding dimensionality. We compare the baseline which uses all attributes, with an SVD dimensionality reduction and a sampling of attributes (we report the mean and standard deviation over 10 samplings). For the SVD CUB results, note the drop at 50 dimensions. We believe this is because the SVD is not guaranteed to choose the most discriminative output dimensions.

	ALE	HLE	AHLE
AWA	37.4	39.0	**43.5**
CUB	**18.0**	12.1	17.0

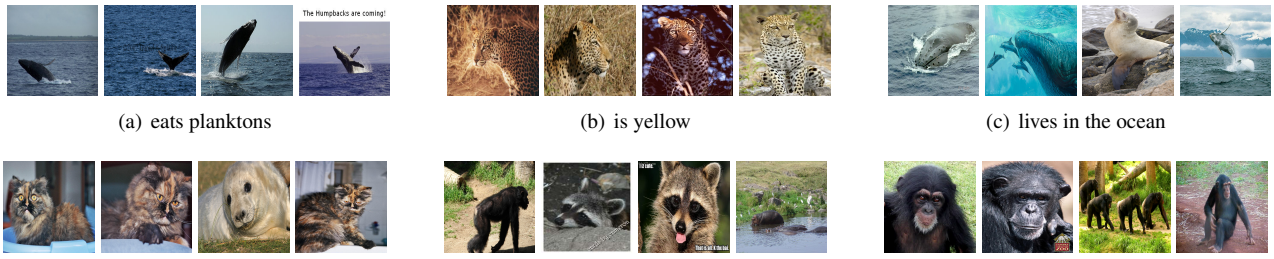
Table 4. Comparison of attributes (ALE) and hierarchies (HLE) for label embedding. We also consider their combination by simple concatenation (AHLE).

Attribute correlation. While correlation in the input space is a well-studied topic, comparatively little work has been done to measure the correlation in the output space. Here, we reduce the output space dimensionality and study the impact on the classification accuracy. We explore two different techniques: Singular Value Decomposition (SVD) and attribute sampling. For SVD, we learn on AWA (resp. CUB) the SVD on the 40×85 (resp. 150×312) $\Phi^{\mathcal{A}}$ matrix and then project the remaining 10 (resp. 50) evaluation classes in this space. For the sampling, we sub-sample a fixed number of attributes and repeat the experiments 10 times with different sub-samplings. We show results in Figure 3.

From these experiments, we can conclude that there is a significant amount of correlation between attributes and that the output space dimensionality can be significantly reduced with little accuracy loss. For instance, on AWA the accuracy drops from 37.4 to 35.7 when reducing from an 85-dim space to a 10-dim space. On CUB the accuracy drops from 18.0 to 17.2 when reducing from a 312-dim space to a 20-dim space. As expected, SVD outperforms a random sampling of the attribute dimensions.

Comparison of ALE and HLE. As mentioned earlier, while attributes can be a useful source of prior information to embed classes, other sources exist. We consider as an alternative the Wordnet hierarchy. We collect from Wordnet the set of ancestors of the 50 AWA (resp. 200 CUB) classes and build a hierarchy with 150 (resp. 299) nodes[5]. We used the $\{0, 1\}$ embedding with ℓ_2-norm.

[5]In some cases, some of the nodes have a single child. We did not clean the automatically obtained hierarchy.



(a) eats planktons (b) is yellow (c) lives in the ocean

(d) is stalker (e) is hunter (f) is smart

Figure 2. Sample attributes recognized with high (*i.e.* >90%) accuracy (top) and low (*i.e.* <50%) accuracy (bottom) by ALE on AWA. For each attribute we show the images ranked highest. Note that a AUC < 50% means that the prediction is worse than random on average.

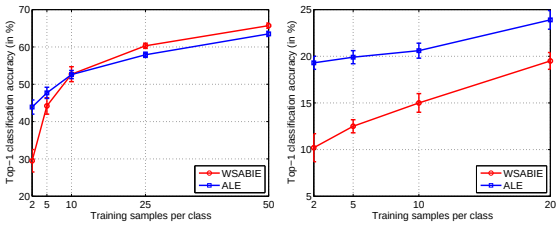


Figure 4. Classification accuracy on AWA (left) and CUB (right) as a function of the number of training samples per class.

We also consider the combination of attributes and hierarchies. We explore different alternatives such as the concatenation of the embeddings or performing CCA on the embeddings. The simpler concatenation alternative always outperformed the more complex CCA and therefore we only report results with the former approach.

Results are shown in Table 4. While on AWA the HLE performs slightly better than ALE, on CUB ALE perform significantly better. Such a behaviour could be expected since the CUB attributes were obtained in a much more controlled and exhaustive way than on AWA. Also, since CUB is a finer-grained dataset than AWA, the CUB hierarchy is much "flatter" than the AWA hierarchy and, therefore, certainly not as informative. On AWA, the combination performs better than attributes or the hierarchy alone while on CUB, there is no improvement through the combination, certainly because the hierarchy adds little additional information. Hence, a class hierarchy can be used as a complementary source of information for "poor-man" attributes.

4.3. Beyond zero-shot learning

We now report results when learning with few examples (shots) and when learning with the full datasets. The goal is to show that, with label embedding, we can combine prior information and labeled data.

Few-shots learning. In these experiments, we assume that we have few (*e.g.* 2, 5, 10, etc.) training samples for each of the 10 AWA (resp. 50 CUB) evaluation classes plus all training samples from the remaining 40 AWA (resp. 150 CUB) classes to learn and cross-validate classifiers. Evaluation is done on the "test" set of the 10 AWA (resp. 50 CUB) classes. We compare ALE with WSABIE [41] which performs label embedding and therefore "shares" samples between classes but does not use prior information. For ALE and WSABIE, W is initialized to the matrix learned in the zero-shot experiments. We show results in Figure 4. On AWA, ALE outperforms WSABIE significantly for a small amount of training data but is outperformed by WSABIE for 25 training samples per class or more. One advantage of WSABIE with respect to ALE is that the embedding space dimensionality can be tuned, thus giving more flexibility when larger amounts of training data become available. On the other hand, on CUB ALE always outperforms WSABIE. Note that the maximum number of training samples per class we used for CUB is 20 because the least populated class has only 42 samples (21 training). As an example, ALE with 2 training samples performs on par with WSABIE with 20 training samples, showing that attributes can compensate for limited training data.

	OVR	WSABIE	ALE	HLE	AHLE
AWA	52.3	49.6	49.7	52.6	**52.9**
CUB	23.4	20.1	20.5	20.1	**23.5**

Table 5. Comparison of different learning algorithms on the full datasets (50 resp. 200 classes). OVR and WSABIE do not use any prior information while ALE, HLE and AHLE do.

Learning and testing on the full datasets. In these experiments, we learn and test the classifiers on the 50 AWA (resp. 200 CUB) classes. We use the "train" set for training and cross-validation and the "test" set to measure accuracy. We compare three embedding techniques: ALE (attributes only), HLE (hierarchy only), AHLE (attributes and hierarchy). We also provide two baselines: a One-Vs-Rest (OVR) binary SVM (which does not consider parameter sharing) and WSABIE (which performs parameter sharing without side information). As can be seen in Table 5, the OVR baseline performs on par with AHLE. We hypothesize that this is because a priori information plays a limited role when training data is plentiful. To test this hypothesis, we experimented with only half of the training data on CUB. In such a case AHLE outperforms OVR (17.6% accuracy *vs.* 16.4%) which seems to validate our hypothesis.

Finally, training the proposed approach is efficient. Using a single processor of a Linux server with 2.4GHz Intel Xeon processors and 32GBs of RAM, it takes approx. 3hrs on CUB to learn the AHLE parameters with the 65,536-dim FVs.

5. Conclusion

We proposed to cast the problem of attribute-based classification as one of label-embedding. This formulation addresses in a principled fashion the limitations of the original DAP model. First, we solve directly the problem at hand (image classification) without introducing an intermediate problem (attribute classification). Second, our model can leverage labeled training data (if available) to update the label embedding, using the attribute embedding as a prior. Third, the label embedding famework is not restricted to attributes and can accommodate other sources of prior information such as class taxonomies. In the zero-shot setting, we demonstrated improved results with respect to DAP. In the few-shots setting, we showed improvements with respect to WSABIE, which learns the label embedding from labeled data but does not leverage prior information.

References

[1] Y. Amit, M. Fink, N. Srebro, and S. Ullman. Uncovering shared structures in multiclass classification. In *ICML*, 2007. 2, 3

[2] S. Bengio, J. Weston, and D. Grangier. Label embedding trees for large multi-class tasks. In *NIPS*, 2010. 2, 3

[3] S. Branson, C. Wah, B. Babenko, F. Schroff, P. Welinder, P. Perona, and S. Belongie. Visual recognition with humans in the loop. In *ECCV*, 2010. 6

[4] K. Chatfield, V. Lempitsky, A. Vedaldi, and A. Zisserman. The devil is in the details: an evaluation of recent feature encoding methods. In *BMVC*, 2011. 2, 5

[5] S. Clinchant, G. Csurka, F. Perronnin, and J.-M. Renders. XRCEs participation to ImageEval. In *ImageEval Workshop at CVIR*, 2007. 5

[6] T. Deselaers and V. Ferrari. Visual and semantic similarity in ImageNet. In *CVPR*, 2011. 2

[7] M. Douze, A. Ramisa, and C. Schmid. Combining attributes and Fisher vectors for efficient image retrieval. In *CVPR*, 2011. 2

[8] A. Farhadi, I. Endres, D. Hoiem, and D. Forsyth. Describing objects by their attributes. *CVPR*, 2009. 1, 2, 3

[9] V. Ferrari and A. Zisserman. Learning visual attributes. In *NIPS*, 2007. 2

[10] B. Geng, L. Yang, C. Xu, and X.-S. Hua. Ranking model adaptation for domain-specific search. *IEEE TKDE*, 2012. 4

[11] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning (2nd Ed.)*. Springer Series in Statistics. Springer, 2008. 3

[12] D. Hsu, S. Kakade, J. Langford, and T. Zhang. Multi-label prediction via compressed sensing. In *NIPS*, 2009. 2

[13] H. Jégou, M. Douze, and C. Schmid. Product quantization for nearest neighbor search. *IEEE TPAMI*, 2011. 2, 5

[14] G. Kulkarni, V. Premraj, S. Dhar, S. Li, Y. choi, A. Berg, and T. Berg. Baby talk: understanding and generating simple image descriptions. In *CVPR*, 2011. 2

[15] N. Kumar, P. Belhummeur, and S. Nayar. FaceTracer: A search engine for large collections of images with faces. In *ECCV*, 2008. 2

[16] C. Lampert, H. Nickisch, and S. Harmeling. Learning to detect unseen object classes by between-class attribute transfer. In *CVPR*, 2009. 1, 2, 3, 4, 5, 6

[17] H. Larochelle, D. Erhan, and Y. Bengio. Zero-data learning of new tasks. In *AAAI*, 2008. 1, 2, 3

[18] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60:91–110, 2004. 5

[19] D. Mahajan, S. Sellamanickam, and V. Nair. A joint learning framework for attribute models and object descriptions. In *ICCV*, 2011. 2

[20] S. Maji and A. Berg. Max-margin additive classifiers for detection. In *ICCV*, 2009. 2

[21] T. Mensink, J. Verbeek, and G. Csurka. Tree-structured CRF models for interactive image labeling. *IEEE TPAMI*, 2012. 2

[22] T. Mensink, J. Verbeek, F. Perronnin, and G. Csurka. Metric learning for large scale image classification: Generalizing to new classes at near-zero cost. In *ECCV*, 2012. 2

[23] V. Ordonez, G. Kulkarni, and T. Berg. Im2Text: Describing images using 1 million captioned photographs. In *NIPS*, 2011. 2

[24] D. Osherson, J. Stern, O. Wilkie, M. Stob, and E. Smith. Default probability. *Cognitive Science*, 1991. 4

[25] M. Palatucci, D. Pomerleau, G. Hinton, and T. Mitchell. Zero-shot learning with semantic output codes. In *NIPS*, 2009. 1, 2, 3, 5

[26] F. Perronnin, J. Sánchez, and T. Mensink. Improving the Fisher kernel for large-scale image classification. In *ECCV*, 2010. 2, 5

[27] M. Rohrbach, M. Stark, and B.Schiele. Evaluating knowledge transfer and zero-shot learning in a large-scale setting. In *CVPR*, 2011. 2

[28] M. Rohrbach, M. Stark, G. Szarvas, I. Gurevych, and B. Schiele. What hepls here – and why? Semantic relatedness for knowledge transfer. In *CVPR*, 2010. 2

[29] M. Saerens, F. Fouss, L. Yen, and P. Dupont. The principal components analysis of a graph, and its relationships to spectral clustering. In *ECML*, 2004. 4

[30] J. Sánchez and F. Perronnin. High-dimensional signature compression for large-scale image classification. In *CVPR*, 2011. 2

[31] V. Sharmanska, N. Quadrianto, and C. H. Lampert. Augmented attribute representations. In *ECCV*, 2012. 2

[32] J. Shawe-Taylor and N. Cristianini. *Kernel Methods for Pattern Analysis*. Cambridge Univ. Press, 2004. 2

[33] B. Siddiquie, R. Feris, and L. Davis. Image ranking and retrieval based on multi-attribute queries. In *CVPR*, 2011. 2

[34] I. Tsochantaridis, T. Joachims, T. Hofmann, and Y. Altun. Large margin methods for structured and interdependent output variables. *JMLR*, 2005. 3, 4

[35] A. Vedaldi and A. Zisserman. Efficient additive kernels via explicit feature maps. In *CVPR*, 2010. 2

[36] A. Vedaldi and A. Zisserman. Sparse kernel approximations for efficient classification and detection. In *CVPR*, 2012. 2

[37] C. Wah, S. Branson, P. Perona, and S. Belongie. Multiclass recognition and part localization with humans in the loop. In *ICCV*, 2011. 2, 4, 6

[38] G. Wang and D. Forsyth. Joint learning of visual attributes, object classes and visual saliency. In *ICCV*, 2009. 2

[39] Y. Wang and G. Mori. A discriminative latent model of object classes and attributes. In *ECCV*, 2010. 2

[40] K. Weinberger and O. Chapelle. Large margin taxonomy embedding for document categorization. In *NIPS*, 2008. 2, 3

[41] J. Weston, S. Bengio, and N. Usunier. Large scale image annotation: Learning to rank with joint word-image embeddings. *ECML*, 2010. 2, 3, 4, 7

[42] J. Weston, O. Chapelle, A. Elisseeff, B. Schölkopf, and V. Vapnik. Kernel dependency estimation. In *NIPS*, 2002. 2

[43] X. Yu and Y. Aloimonos. Attribute-based transfer learning for object categorization with zero or one training example. In *ECCV*, 2010. 2